

Extract Method Refactoring in C# with Lambda Expression

Takuto Kawamoto Yoshiki Higo

The University of Osaka, Japan

Code Reuse

The practice of using existing code fragments in other contexts.

Good Approaches: Extract Method, Inheritance,...

Good Maintainability

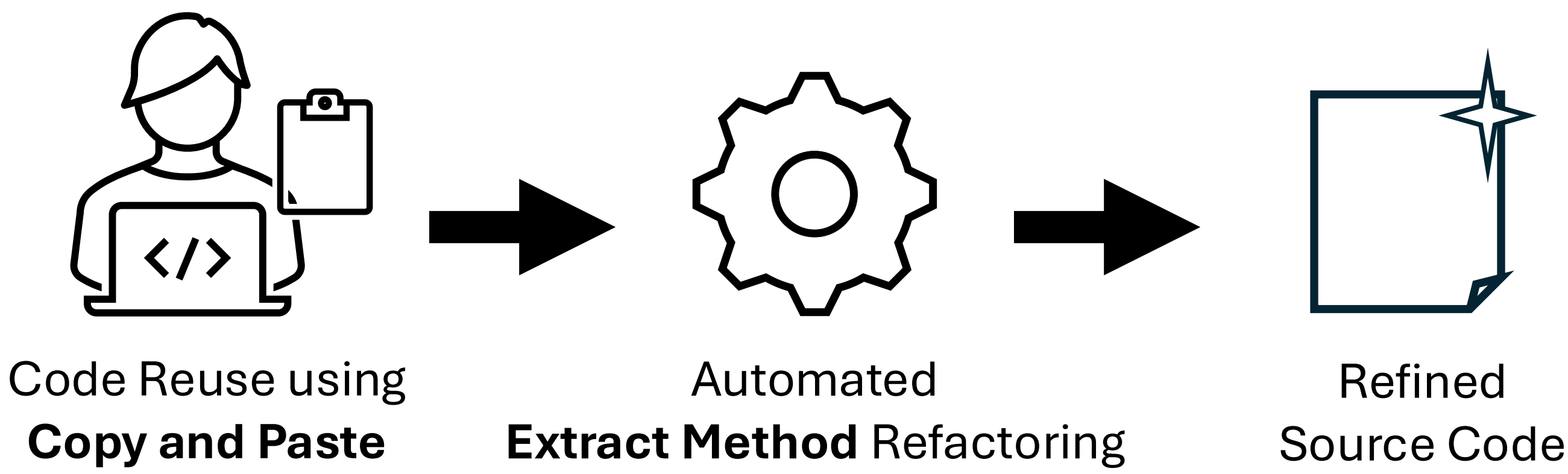
Complicated Work, Risk of Bugs Being Introduced

Bad Approach: Copy and Paste

Fast, Easy

Occurrence of Code Clones, Reduced maintainability

Combined Approach



Developers can focus on simple implementation, while the system handles the removal of code clone.

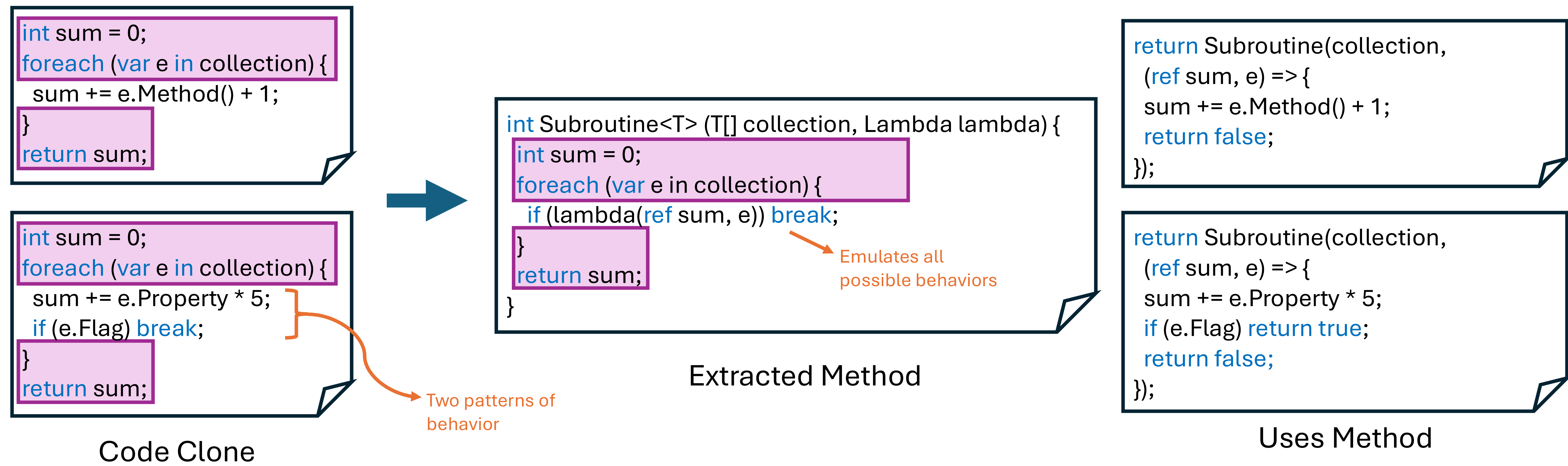
Target Code Clone

- There are some gaps between the code fragments.
- The gap may span multiple statements.

Motivations

- In existing studies,
- The target programming language is limited to Java.
 - Jump statements are not taken into account.

e.g. `return`, `break`, `continue` statement



Our Automated Refactoring Steps

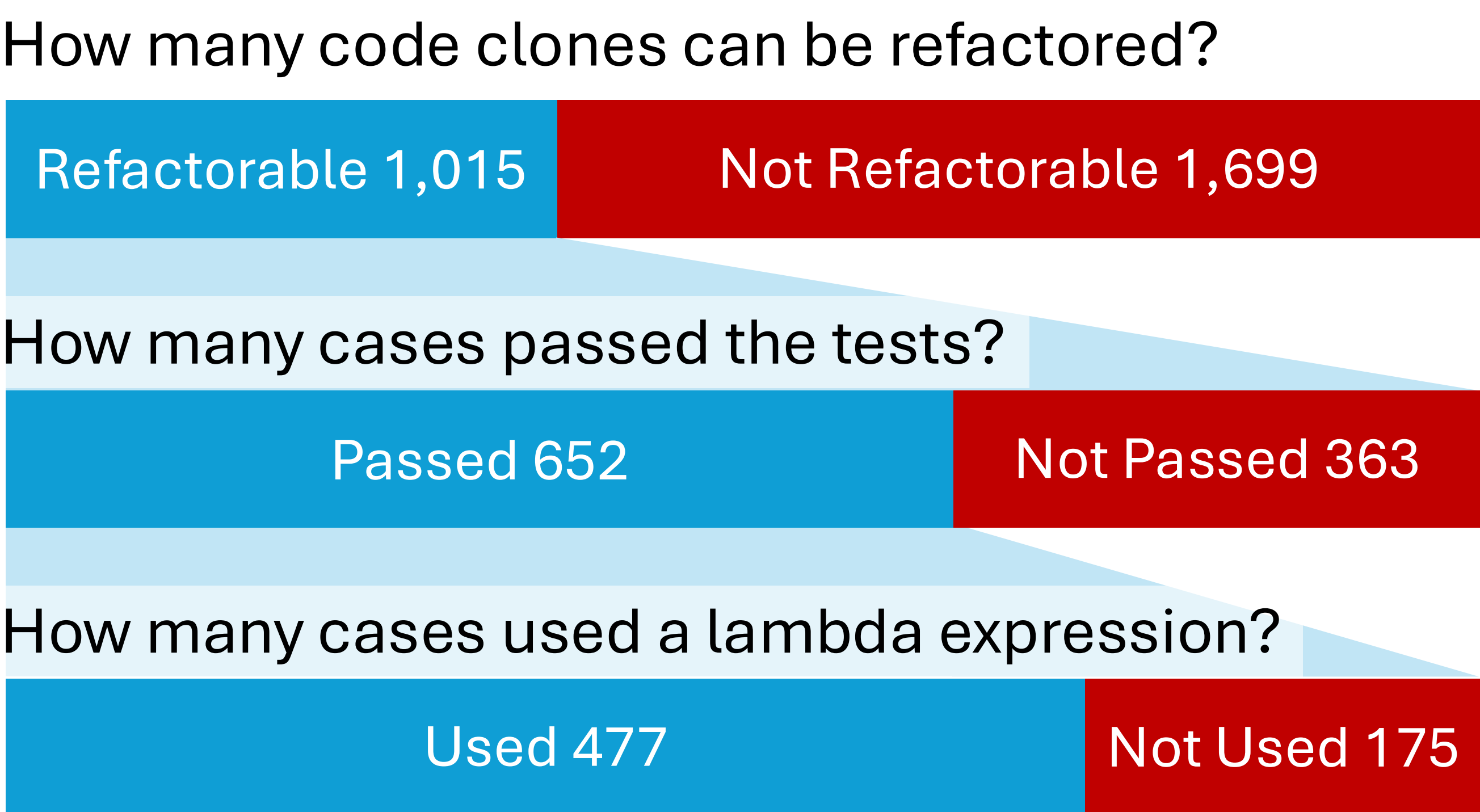
1. Statement Mapping
2. Variable Mapping
3. Exit Block Checking
4. Argument Determination
5. Source Code Generation

Our process assesses if a clone pair is refactorable, and if so, apply the refactoring.

Evaluation

1. Collect projects from GitHub
2. Detect code clones by NiCad
3. Apply the proposed automated refactoring
4. Verify behavior via existing tests

We collected 2,714 code clones from 28 C# projects.



Conclusion

- 30.2% of the discovered code clones can be refactored.
- 73.2% of verified cases used lambda expressions.

Future Work

- Improve implementation
- Automate other Code Reuse approaches